پروژه دوم علوم اعصاب محاسباتی

- صورت پروژه در این آدرس قابل مشاهده است.
- با توجه به دشواری حفظ ساختار صورت پروژه در گزارش، آن ساختار نادیده گرفته شده و مطالب با ساختاری مناسب برای دنبال کردن نمودارها و مطالب منظم شده‌اند؛ با اینحال تمام مطالب خواسته شده در صورت پروژه، در این گزارش پوشانده شده‌اند.
- در فاز قبلی این پروژه، رفتار نورونی مدل LIF به تفضیل مورد بررسی قرار گرفت. گزارش مذکور در این آدرس قابل مشاهده است.

0. فهرست مطالب

In [1]:
```
from cnsproject.network.neural_populations import LIFPopulation,ELIFPopulatio
from cnsproject.network.monitors import Monitor
from cnsproject.plotting.plotting import Plotter
from cnsproject.utils import *
import matplotlib.pyplot as plt
import torch
```

بجز ابزار شبیه‌سازی (که import شده‌اند)، تابع پایین در این تمرین خاص، برای شبیه‌سازی و مقایسه نورون‌ها در کنار هم به ما کمک خواهد کرد. همچنین در این تمرین، هر شبیه‌سازی را به مدت 200ms ادامه خواهیم داد.

In [64]:
```
time=200 #ms
def neuron_behaviour(neuron_cls, I, p, time=time, postfix='', dt=1., name='',
                     threshold='firing_threshold', w=False, **args):
    neuron = neuron_cls((1,), dt=dt, **args)
    monitor = Monitor(neuron, state_variables=["s", "u"] if not w else ["s",
    neuron.refractory_and_reset()
    monitor.simulate(neuron.forward, {'I': I})
    p.monitor = monitor
    p.neuron_spike('s'+postfix, x_vis=False, y_label='spikes', title=name)
    p.neuron_voltage('u'+postfix, x_vis=False, y_label="u (mV)", x_label='',
    if w:
        p.adaptation_current_dynamic('w'+postfix, y_label="w (mV)", x_vis=Fal
    p.current_dynamic('i'+postfix, I=I, y_label="I (mA)", repeat_till=time)
```

1. مدل ELIF

برای پیاده‌سازی این مدل نورونی، کافی است اختلاف پتانسیل نورون را در هر گام طبق رابطه زیر به روزرسانی کنیم:

$$U(t+\Delta) = U(t) - \frac{\Delta}{\tau}[(U(t) - U_{rest}) - \Delta_T e^{\frac{U(t)-U_{firing-threshold}}{\Delta_T}} - R.I(t)]$$

$$if \;\; U(t) > U_{spike-threshold} : U(t) = 0 \quad and \quad spike-on!$$

از آنجایی که این مدل توسعه‌ای بر مدل LIF است، پس انتظاراتی که از مدل LIF داشتیم، اینجا نیز مورد انتظار هستند:
1- با افزایش میزان جریان ورودی، اختلاف پتانسیل مثبت‌تر و درنتیجه فرکانس spike خروجی بیشتری شاهد هستیم.
2- با افزایش میزان مقاوت (R)، میزان تأثیرپذیری اختلاف پتانسیل نورون از جریان ورودی بیشتر می‌شود. به زبان ساده‌تر، با دریافت جریان ورودی، اختلاف پتانسیل با سرعت بیشتری افزایش پیدا می‌کند (مثبت می‌شود) و درنتیجه فرکانس spike خروجی افزایش پیدا می‌کند.
3- با افزایش τ، به صورت کلی سرعت تغییرات اختلاف پتانسیل کاهش پیدا می‌کند (لختی میزانی اختلاف پتانسیل بیشتر می‌شود).
4- با کاهش مقدار اختلاف $U_{spike-threshold}$ و U_{rest}، شاهد فرکانس بیشتری در spikeها خواهیم بود.

علاوه بر انتظارات فوق، انتظارات زیر نیز وجود دارند:
5- با افزایش Δ_T، سرعت افزایش پتانسیل زیاد شده و درنتیجه، فرکانس spike افزایش پیدا می‌کند.

6- با افزایش زیاد Δ_T، نمودار رشد پتانسیل به صورت کلی بالای محور افق قرار می‌گیرد و درنتیجه، پتانسیل نورون بدون نیاز به جریان ورودی افزایش پیدا می‌کند. با توجه به نشتی پتانسیل نورونی، در نتیجه این افزایش و کاهش همزمان، نقطه همگرایی پتانسیلی بالا resting potential خواهیم داشت. اگر Δ_T به اندازه‌ای بزرگ باشد که این نقطه همگرایی بالای firing threshold قرار بگیرد، نورون بدون جریان ورودی spike خواهد زد.
7- نورون در زمان firing باید رفتار انفجاری (افزایش ناگهانی) داشته باشد.

با توجه به بدیهی بودن نتیجه ۴، آن را بررسی نخواهیم کرد.

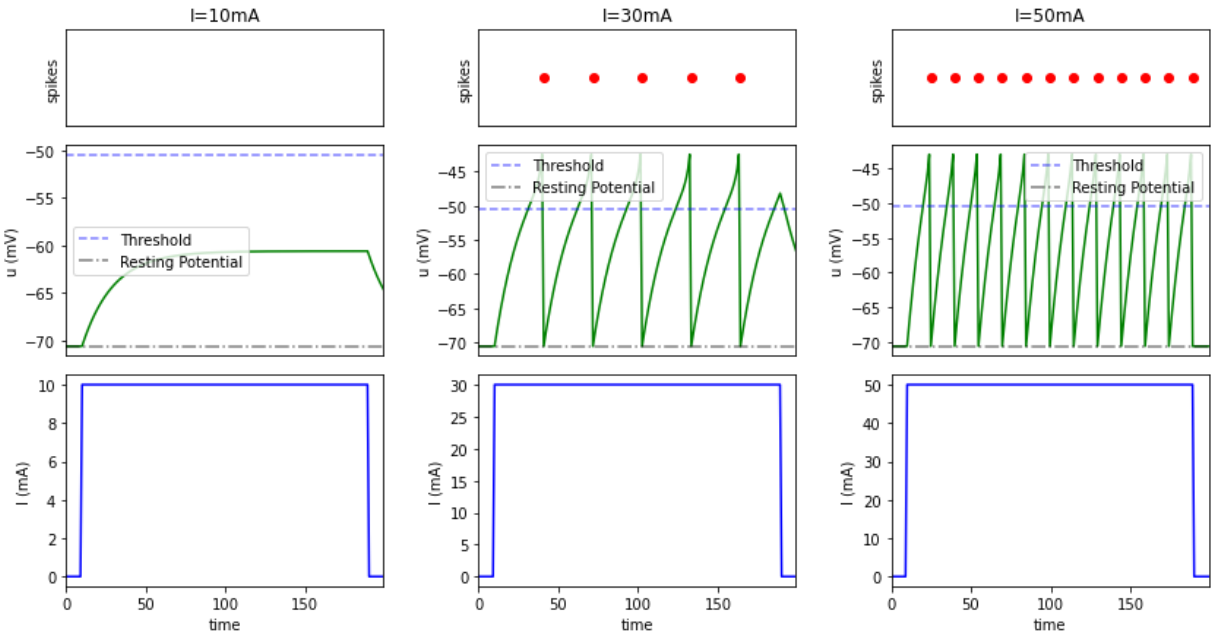
A.1. جریان ورودی ثابت

a.A.1. بررسی رفتار مدل با دامنه‌های متفاوت جریان ورودی

In [65]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

Ion = 10
I = step_function(time, 10, val1=Ion) - step_function(time, time-10, val1=Ior
neuron_behaviour(ELIFPopulation, I, p, postfix='1', name="I=10mA")
Ion = 30
I = step_function(time, 10, val1=Ion) - step_function(time, time-10, val1=Ior
neuron_behaviour(ELIFPopulation, I, p, postfix='2', name="I=30mA")
Ion = 50
I = step_function(time, 10, val1=Ion) - step_function(time, time-10, val1=Ior
neuron_behaviour(ELIFPopulation, I, p, postfix='3', name="I=50mA")

p.show()
```



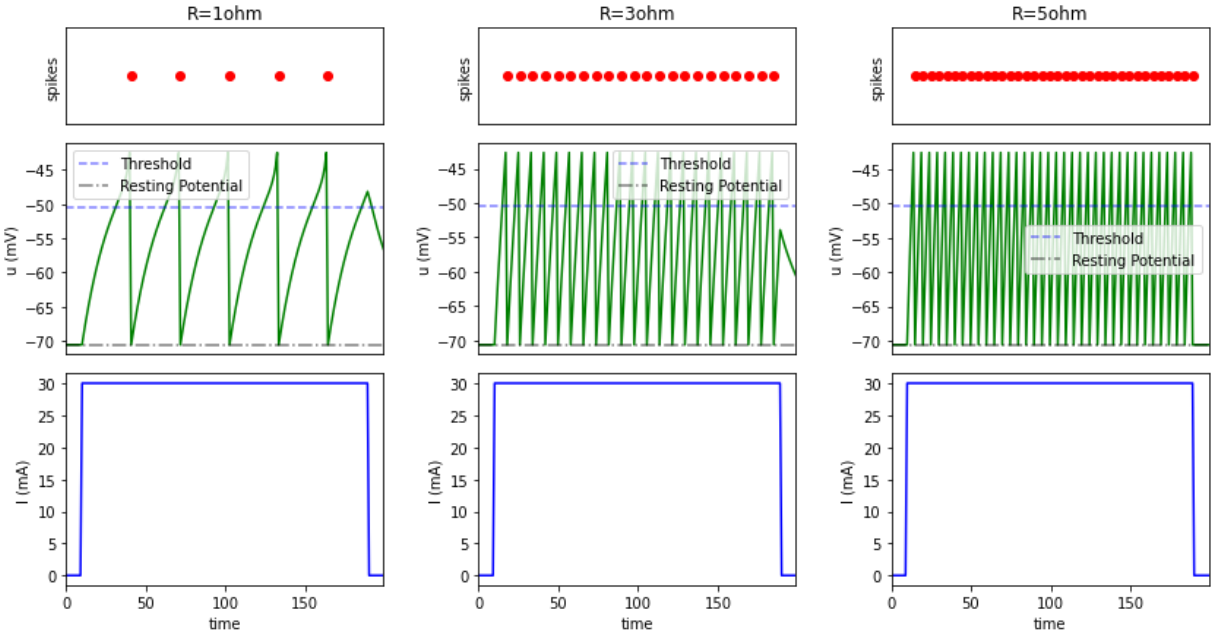
نتیجه مطابق انتظار اول ما از مدل است. همچنین رفتار انفجاری را مطابق بند ۷ از انتظارات خود از مدل، مشاهده می‌کنیم.

b.A.1. بررسی رفتار مدل با مقاوت‌های متفاوت

In [28]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
Ion = 30
I = step_function(time, 10, val1=Ion) - step_function(time, time-10, val1=Ior

neuron_behaviour(ELIFPopulation, I, p, R=1, postfix='1', name="R=1ohm")
neuron_behaviour(ELIFPopulation, I, p, R=3, postfix='2', name="R=3ohm")
neuron_behaviour(ELIFPopulation, I, p, R=5, postfix='3', name="R=5ohm")

p.show()
```



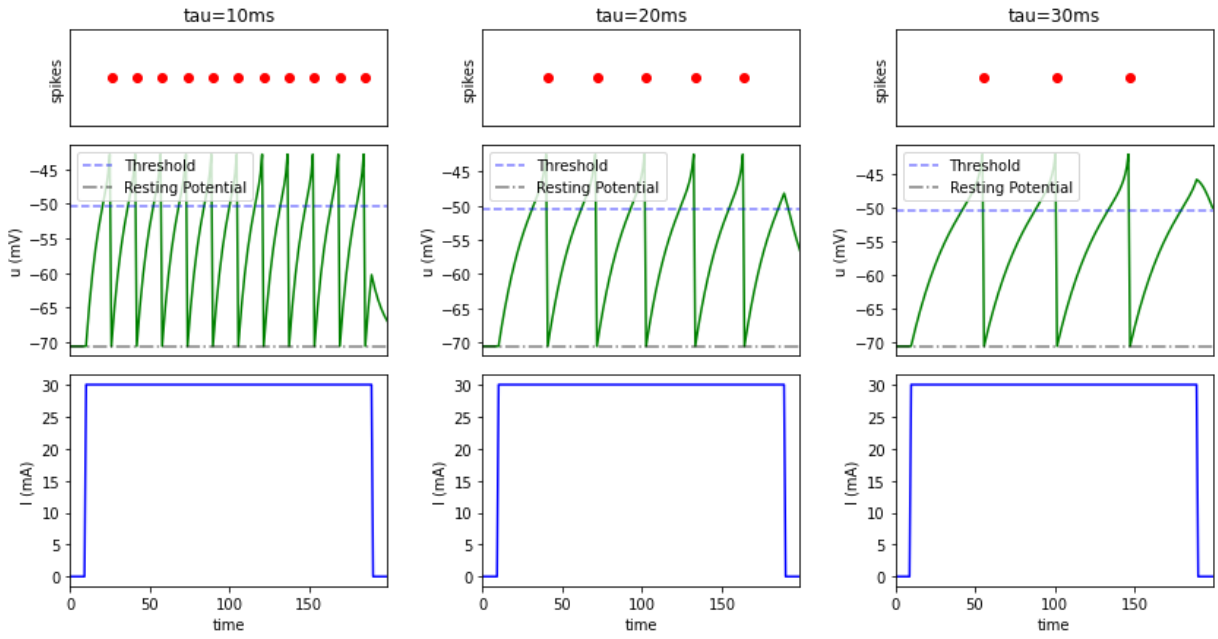
نتیجه مطابق انتظار دوم ما از مدل است.

c.A.1. بررسی رفتار مدل با ثابت زمانی(τ)های متفاوت

In [42]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
Ion = 30
I = step_function(time, 10, val1=Ion) - step_function(time, time-10, val1=Ior

neuron_behaviour(ELIFPopulation, I, p, tau=10, postfix='1', name="tau=10ms")
neuron_behaviour(ELIFPopulation, I, p, tau=20, postfix='2', name="tau=20ms")
neuron_behaviour(ELIFPopulation, I, p, tau=30, postfix='3', name="tau=30ms")

p.show()
```



نتیجه مطابق انتظار سوم ما از مدل است.

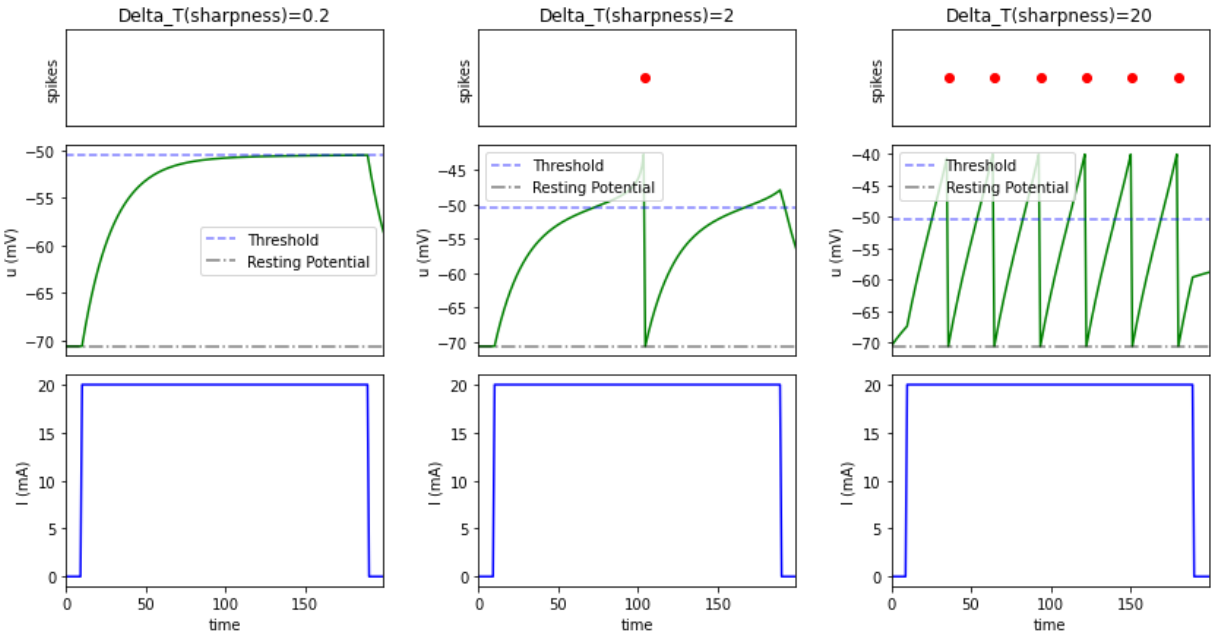
d.A.1. بررسی رفتار مدل با Δ_Tهای متفاوت

i.d.A.1. با جریان ورودی

In [43]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
Ion = 20
I = step_function(time, 10, val1=Ion) - step_function(time, time-10, val1=Ior

neuron_behaviour(ELIFPopulation, I, p, sharpness=.2, postfix='1', name="Delta
neuron_behaviour(ELIFPopulation, I, p, sharpness=2, postfix='2', name="Delta_
neuron_behaviour(ELIFPopulation, I, p, sharpness=20, postfix='3', name="Delta

p.show()
```



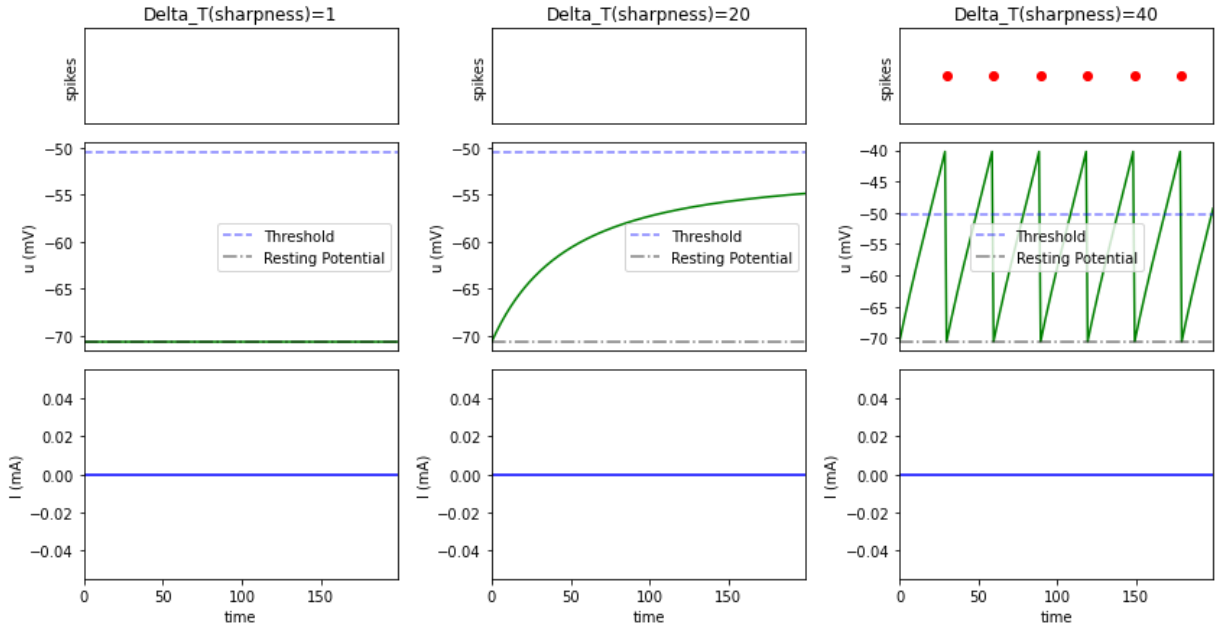
نتیجه مطابق انتظار پنجم ما از مدل است.

ii.d.A.1. بدون جریان ورودی

In [44]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
I = [0]

neuron_behaviour(ELIFPopulation, I, p, sharpness=1, postfix='1', name="Delta_
neuron_behaviour(ELIFPopulation, I, p, sharpness=20, postfix='2', name="Delta
neuron_behaviour(ELIFPopulation, I, p, sharpness=40, postfix='3', name="Delta

p.show()
```



نتیجه مطابق انتظار ششم ما از مدل است.

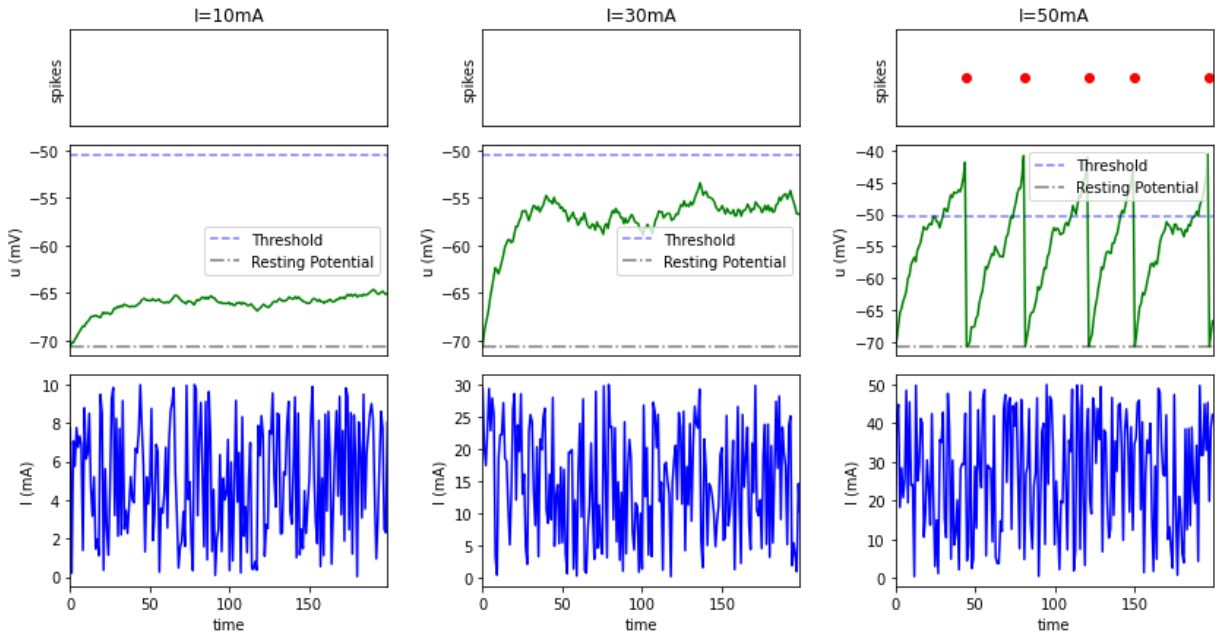
B.1. جریان ورودی تصادفی

a.B.1. بررسی رفتار مدل با دامنه‌های متفاوت جریان ورودی

In [45]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

I = torch.rand(time,1)*10
neuron_behaviour(ELIFPopulation, I, p, postfix='1', name="I=10mA")
I = torch.rand(time,1)*30
neuron_behaviour(ELIFPopulation, I, p, postfix='2', name="I=30mA")
I = torch.rand(time,1)*50
neuron_behaviour(ELIFPopulation, I, p, postfix='3', name="I=50mA")

p.show()
```



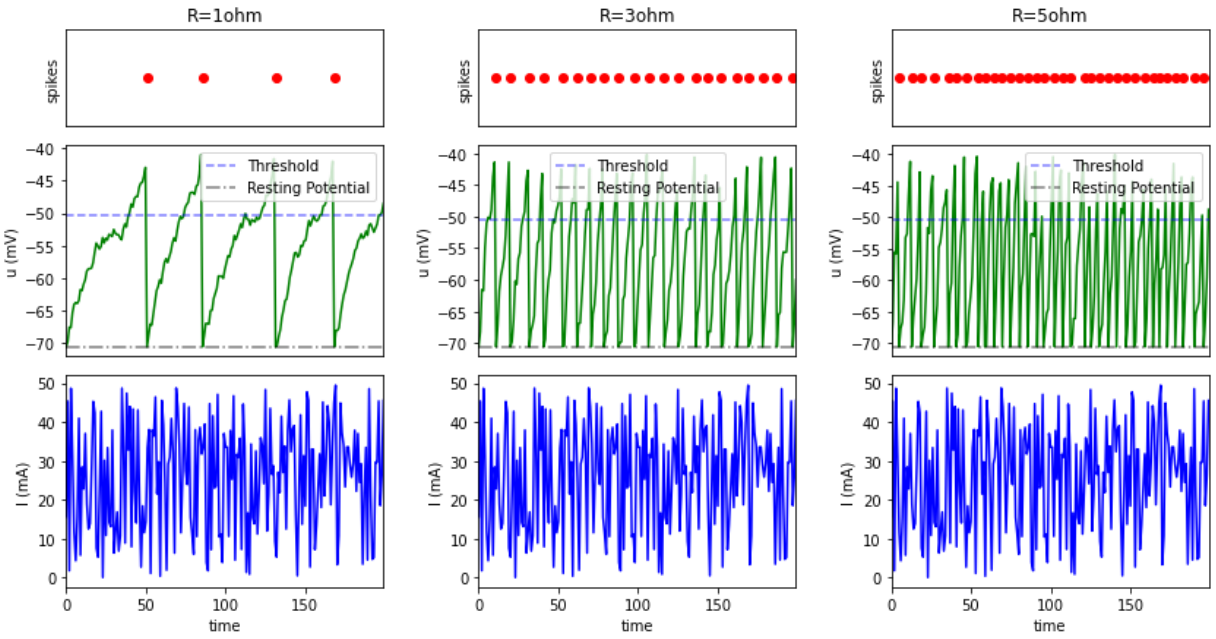
نتیجه مطابق انتظار اول ما از مدل است. همچنین رفتار انفجاری را مطابق بند ۷ از انتظارات خود از مدل، مشاهده می‌کنیم.

b.B.1. بررسی رفتار مدل با مقاوت‌های متفاوت

In [46]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
I = torch.rand(time,1)*50
```

```
neuron_behaviour(ELIFPopulation, I, p, R=1, postfix='1', name="R=1ohm")
neuron_behaviour(ELIFPopulation, I, p, R=3, postfix='2', name="R=3ohm")
neuron_behaviour(ELIFPopulation, I, p, R=5, postfix='3', name="R=5ohm")

p.show()
```



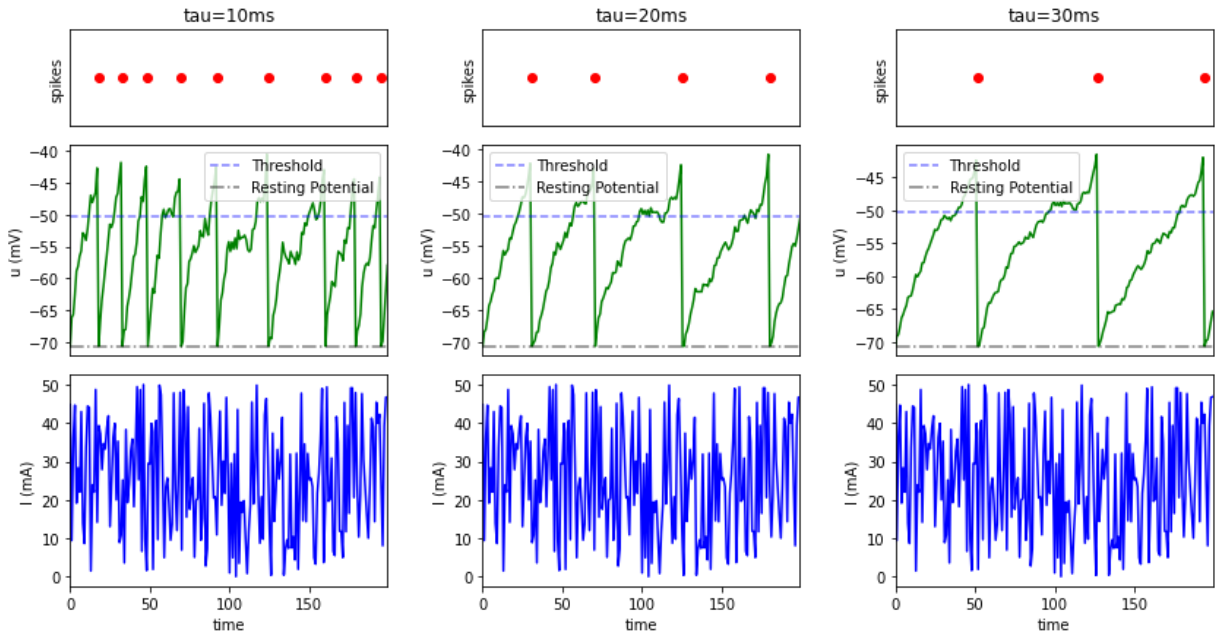
نتیجه مطابق انتظار دوم ما از مدل است.

c.B.1. بررسی رفتار مدل با ثابت زمانی(τ)های متفاوت

In [47]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
I = torch.rand(time,1)*50

neuron_behaviour(ELIFPopulation, I, p, tau=10, postfix='1', name="tau=10ms")
neuron_behaviour(ELIFPopulation, I, p, tau=20, postfix='2', name="tau=20ms")
neuron_behaviour(ELIFPopulation, I, p, tau=30, postfix='3', name="tau=30ms")

p.show()
```



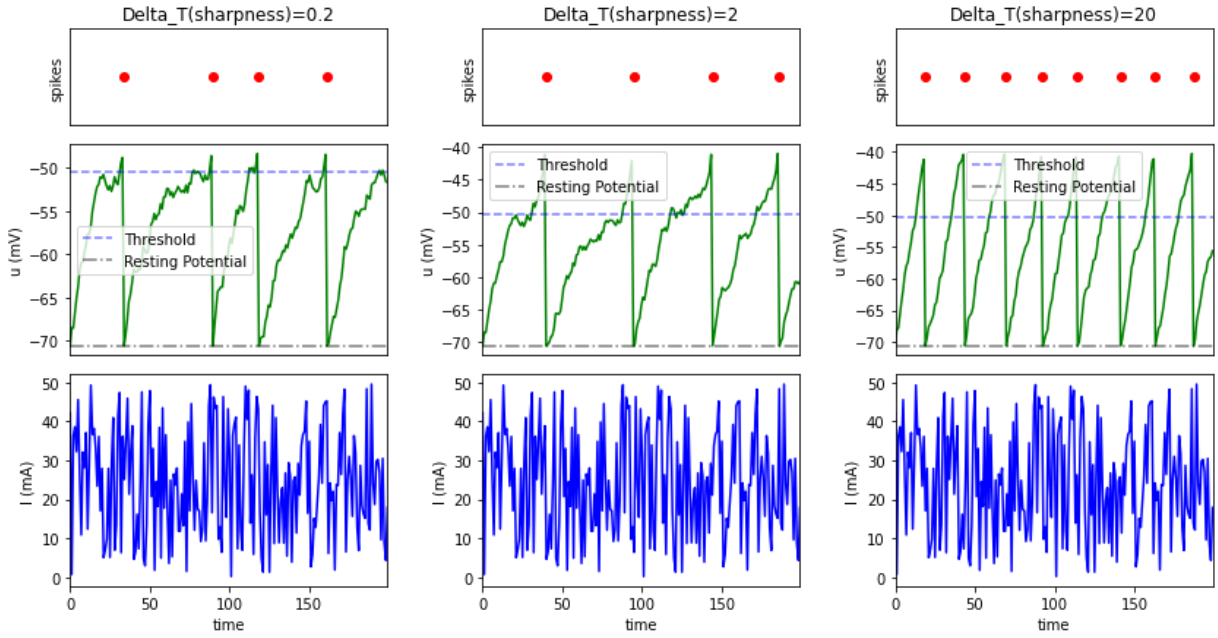
نتیجه مطابق انتظار سوم ما از مدل است.

d.B.1. بررسی رفتار مدل با Δ_Tهای متفاوت

In [48]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
I = torch.rand(time,1)*50

neuron_behaviour(ELIFPopulation, I, p, sharpness=.2, postfix='1', name="Delta
neuron_behaviour(ELIFPopulation, I, p, sharpness=2, postfix='2', name="Delta_
neuron_behaviour(ELIFPopulation, I, p, sharpness=20, postfix='3', name="Delta

p.show()
```



نتیجه مطابق انتظار پنجم ما از مدل است.

2. مدل AELIF

برای پیاده‌سازی این مدل نورونی، کافی است اختلاف پتانسیل نورون را در هر گام طبق رابطه زیر به روزرسانی کنیم:

$$U(t+\Delta) = U(t) - \frac{\Delta}{\tau}[(U(t) - U_{rest}) - \Delta_T e^{\frac{U(t)-U_{firing-threshold}}{\Delta_T}} + R.W(t) - R.I(t)]$$

$$W(t+\Delta) = W(t) + \frac{\Delta}{\tau_w}[a(U(t) - U_{rest}) - W(t) + b\tau_w \sum_{t^f} \delta(t - t^f)]$$

$$if \;\; U(t) > U_{spike-threshold} : U(t) = 0 \quad and \quad spike - on!$$

از آنجایی که این مدل توسعه‌ای بر مدل ELIF است، پس انتظاراتی که از مدل ELIF داشتیم، اینجا نیست مورد انتظار هستند. علاوه بر این انتظارات، انتظارات زیر نیز وجود دارند:
8- با حفظ جریان ورودی در مقادیر بالا به مدت طولانی، با توجه به بزرگ ماندن مقدار $u - u_{rest}$، مقدار w بزرگ شده و در نتیجه مقدار پتانسیل کاهش پیدا می‌کند.
9- پس از مشاهده چند spike متناوب، با توجه به اثر w، مقدار پتانسیل و درنتیجه فرکانس spike در ادامه کاهش پیدا می‌کند.
10- درصورت قطع ناگهانی جریان ورودی، با توجه به غالب شدن ترم $R.W$ نسبت به ترم $R.I$، شاهد افت پتانسیل شدید خواهیم بود. متناسب با مقدار سابق جریان ورودی، مدت زمان حفظ جریان مذکور و پارامترهای مرتبط با w، ممکن است این افت پتانسیل شدت‌های متفاوتی داشته باشد و ممکن است باعث شود اختلاف پتانسیل نورون مقدار کم یا زیادی از resting potential کمتر شود. پس از این رویداد، با توجه به اینکه $u - u_{rest}$ مقداری منفی پیدا می‌کند، از این پس اثر w مثبت بوده و اختلاف پتانسیل به سرعت افزایش پیدا می‌کند تا دوباره از مرز resting potential افزایش پیدا کند. به همین صورت، می‌بایستی شاهد یک نوسان میرا در پتانسیل نورون باشیم. اگر این افت و شدت پتانسیل به اندازه کافی بزرگ باشد (متاثر از اختلاف پتانسیل سابق و مقادیر پارامترها)، ممکن است منجر به spike شود.
11- با افزایش مقدار τ_w، لختی w زیاد می‌شود. درنتیجه، adaptivity دیرتر رخ خواهد داد و ماندگارتر خواهد بود.
12- با افزایش پارامتر b، تاثیر پذیری w و درنتیجه adaptivity از مشاهده spikeهای اخیر بیشتر خواهد شد.
13- با افزایش پارامتر a، تاثیر پذیری w و درنتیجه adaptivity از اختلاف پتانسیل بالا در طول زمان بیشتر خواهد شد.

A.2. جریان ورودی ثابت

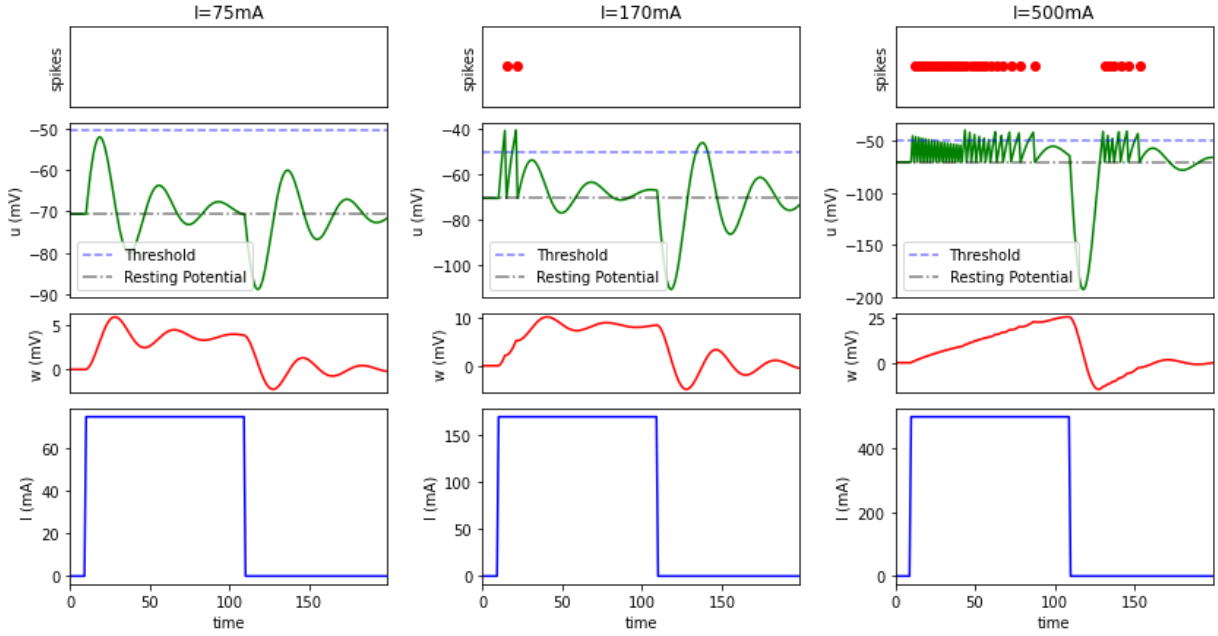
a.A.2. بررسی رفتار مدل با دامنه‌های متفاوت جریان ورودی

In [66]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

Ion = 75
I = step_function(time, 10, val1=Ion) - step_function(time, time-90, val1=Ior
```

```
neuron_behaviour(AELIFPopulation, I, p, postfix='1', name="I=75mA", w=True)
Ion = 170
I = step_function(time, 10, val1=Ion) - step_function(time, time-90, val1=Ior
neuron_behaviour(AELIFPopulation, I, p, postfix='2', name="I=170mA", w=True)
Ion = 500
I = step_function(time, 10, val1=Ion) - step_function(time, time-90, val1=Ior
neuron_behaviour(AELIFPopulation, I, p, postfix='3', name="I=500mA", w=True)

p.show()
```



بدون هیچ توضیح اضافه‌ای، نمودارهای بالا کاملا گویای انتظارات ۱،۷،۸،۹ و ۱۰ از مدل می‌باشند:

۱- از چپ به راست، مشاهده می‌کنیم که با افزایش شدت جریان ورودی، فرکانس spike خروجی افزایش پیدا می‌کند.

۷- در دو نمودار سمت راست می‌بینیم که با گذر پتانسیل نورون از firing threshold، این پتانسیل رفتار انفجارگونه از خود نشان می‌دهد.

۸- نمودار سمت چپ نشان می‌دهد که با بالا ماندن جریان ورودی، adaptivity رخ داده و با افزایش w، مقدار پتانسیل کاهش پیدا می‌کند و شاهد spike نخواهیم بود.

۹- نمودار میانی به خوبی نشان می‌دهد که مشاهده‌ی یک spike در خروجی، باعث افزایش زیاد w می‌شود و فرکانس spike خروجی کاهش می‌یابد. کاهش فرکانس spike خروجی در نمودار سمت راست نیز به خوبی قابل مشاهده است.

۱۰- مطابق با استدلال بیان شده در بخش توضیحات مدل AELIF، مشاهده می‌کنیم که با قطع جریان ورودی (که جریان شدیدی است)، پتانسیل نورون رفتاری نوسانی و میرا از خود نشان می‌دهد. در نمودار سمت راست می‌بینیم که این نوسان آنقدر شدید بوده که باعث ایجاد چندین spike شده است. دلیل این رخداد، شدت بسیار زیاد جریان ورودی سابق در این آزمایش است که خلأ آن به شدت موثر حاضر می‌شود.

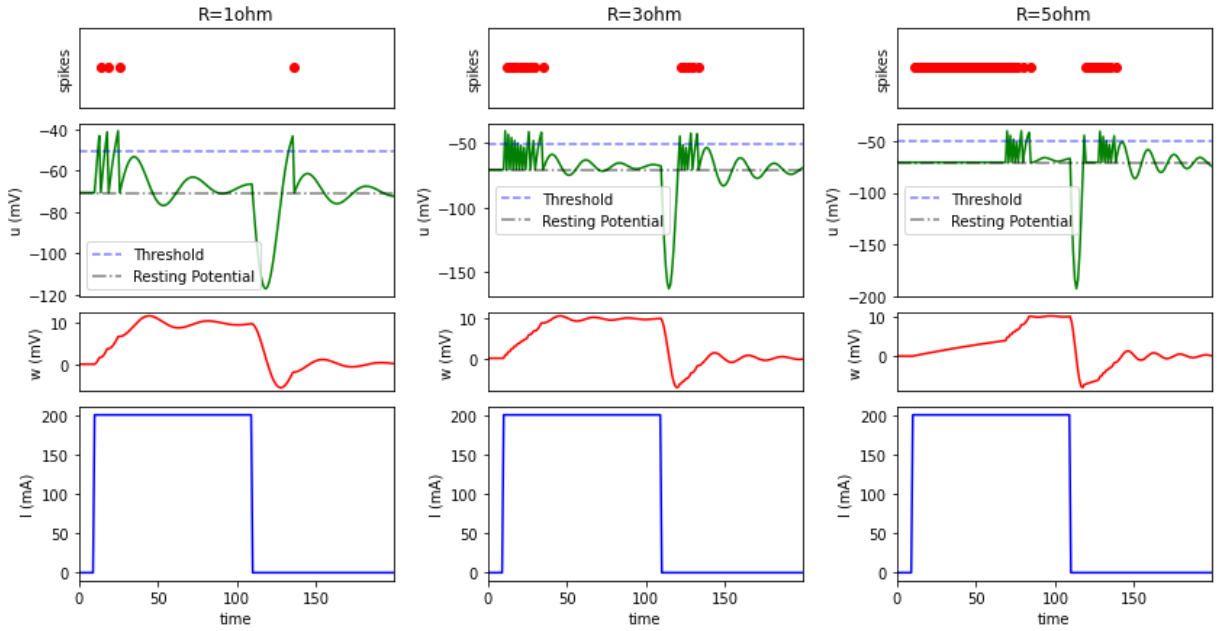
b.A.2. بررسی رفتار مدل با مقاوت‌های متفاوت

In [67]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
```

```
Ion = 200
I = step_function(time, 10, val1=Ion) - step_function(time, time-90, val1=Ior

neuron_behaviour(AELIFPopulation, I, p, R=1, postfix='1', name="R=1ohm", w=Tr
neuron_behaviour(AELIFPopulation, I, p, R=3, postfix='2', name="R=3ohm", w=Tr
neuron_behaviour(AELIFPopulation, I, p, R=5, postfix='3', name="R=5ohm", w=Tr

p.show()
```



نتیجه مطابق انتظار دوم ما از مدل است.

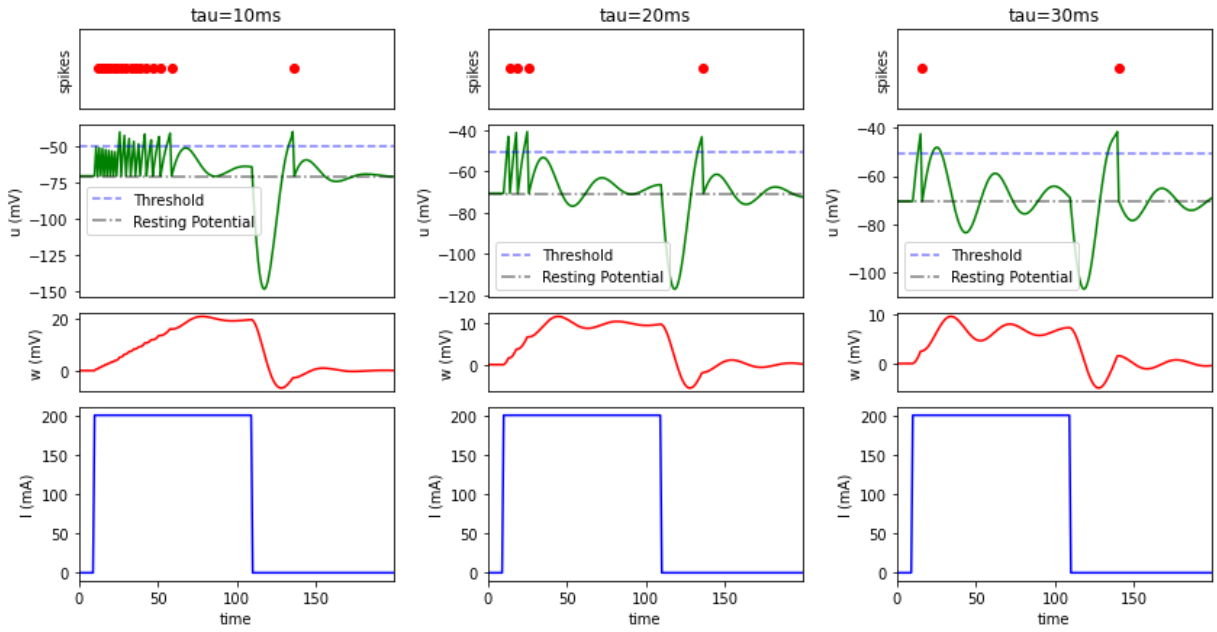
c.A.2. بررسی رفتار مدل با ثابت زمانی(τ)های متفاوت

In [68]:

```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
Ion = 200
I = step_function(time, 10, val1=Ion) - step_function(time, time-90, val1=Ior

neuron_behaviour(AELIFPopulation, I, p, tau=10, postfix='1', name="tau=10ms",
neuron_behaviour(AELIFPopulation, I, p, tau=20, postfix='2', name="tau=20ms",
neuron_behaviour(AELIFPopulation, I, p, tau=30, postfix='3', name="tau=30ms",

p.show()
```



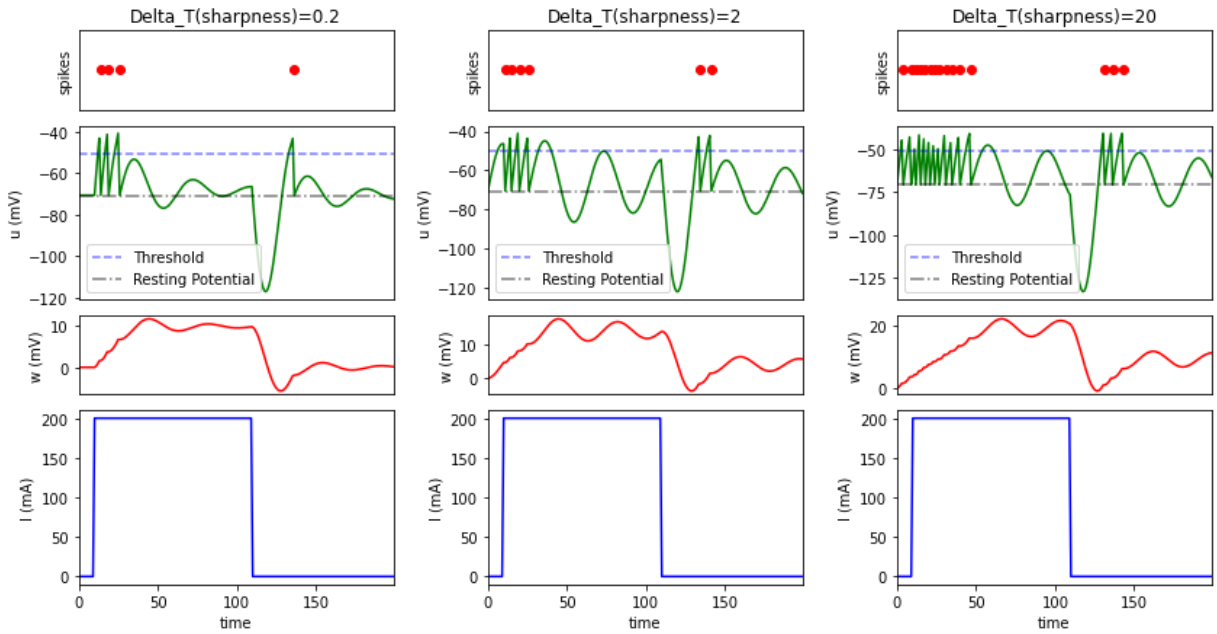
نتیجه مطابق انتظار سوم ما از مدل است.

d.A.2. بررسی رفتار مدل با Δ_Tهای متفاوت

In [71]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
Ion = 200
I = step_function(time, 10, val1=Ion) - step_function(time, time-90, val1=Ior

neuron_behaviour(AELIFPopulation, I, p, sharpness=2, postfix='1', name="Delta
neuron_behaviour(AELIFPopulation, I, p, sharpness=100, postfix='2', name="Del
neuron_behaviour(AELIFPopulation, I, p, sharpness=200, postfix='3', name="Del

p.show()
```



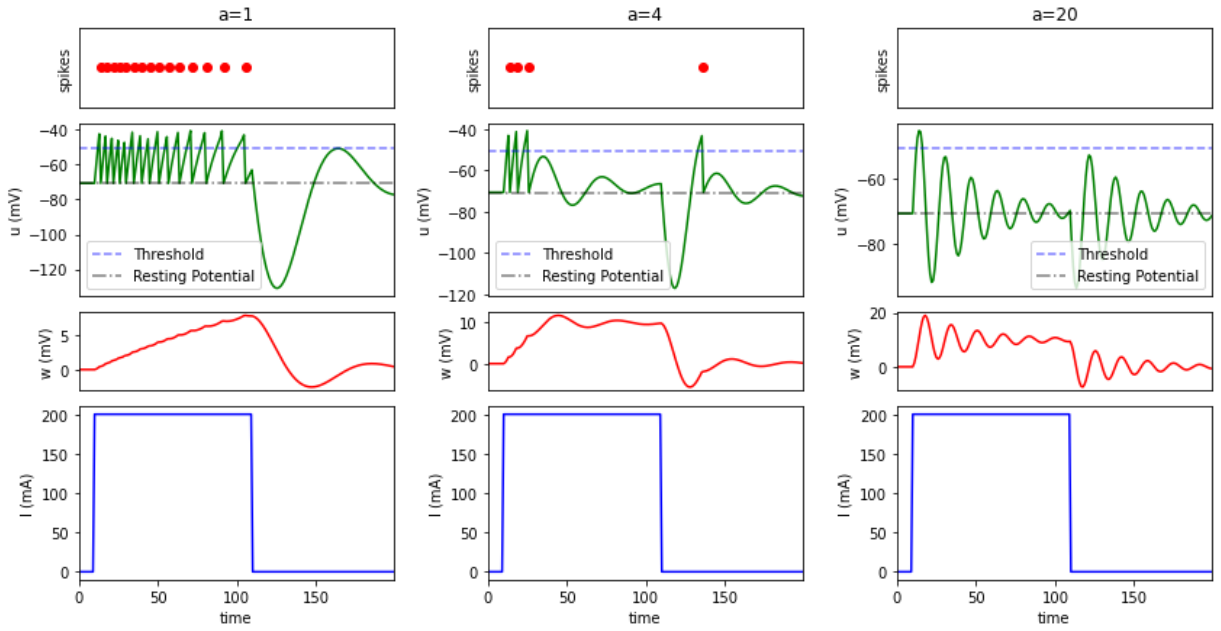
نتیجه مطابق انتظار پنجم ما از مدل است.

e.A.2. بررسی رفتار مدل با مقادیر متفاوت پارامتر a

In [73]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
Ion = 200
I = step_function(time, 10, val1=Ion) - step_function(time, time-90, val1=Ior

neuron_behaviour(AELIFPopulation, I, p, a_w=1, postfix='1', name="a=1", w=Tru
neuron_behaviour(AELIFPopulation, I, p, a_w=4, postfix='2', name="a=4", w=Tru
neuron_behaviour(AELIFPopulation, I, p, a_w=20, postfix='3', name="a=20", w=T

p.show()
```



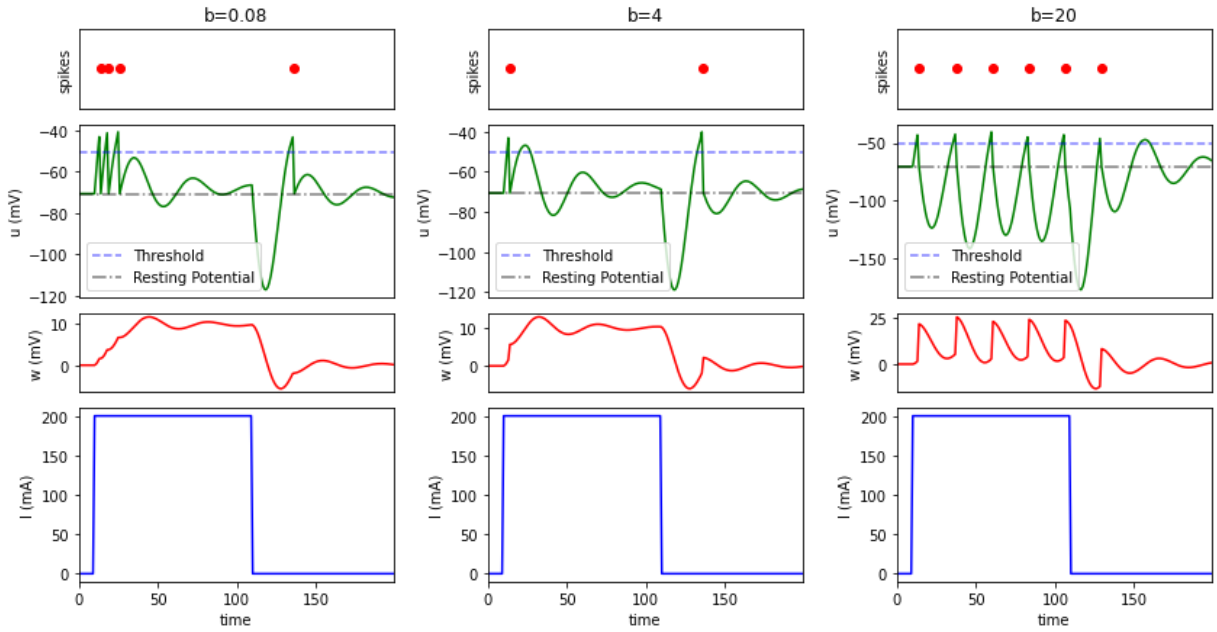
نتیجه مطابق انتظار ۱۳ام ما از مدل است. مدل با a بزرگ‌تر، adaptivity بیشتری از خود نشان داده و درنتیجه فرکانس spike خروجی کاهش پیدا می‌کند.

f.A.2. بررسی رفتار مدل با مقادیر متفاوت پارامتر b

In [82]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
Ion = 200
I = step_function(time, 10, val1=Ion) - step_function(time, time-90, val1=Ior
```

```
neuron_behaviour(AELIFPopulation, I, p, b_w=.08, postfix='1', name="b=0.08",
neuron_behaviour(AELIFPopulation, I, p, b_w=4, postfix='2', name="b=4", w=Tru
neuron_behaviour(AELIFPopulation, I, p, b_w=20, postfix='3', name="b=20", w=T

p.show()
```



نتیجه مطابق انتظار ۱۲ام ما از مدل است. مقدار w تاثیرپذیری بیشتری نسبت به مشاهده spike دارد و به هنگام رخداد spike، به یکباره و به شدت افزایش پیدا می‌کند. همین باعث adaptivity بیشتر در برابر spike می‌شود.

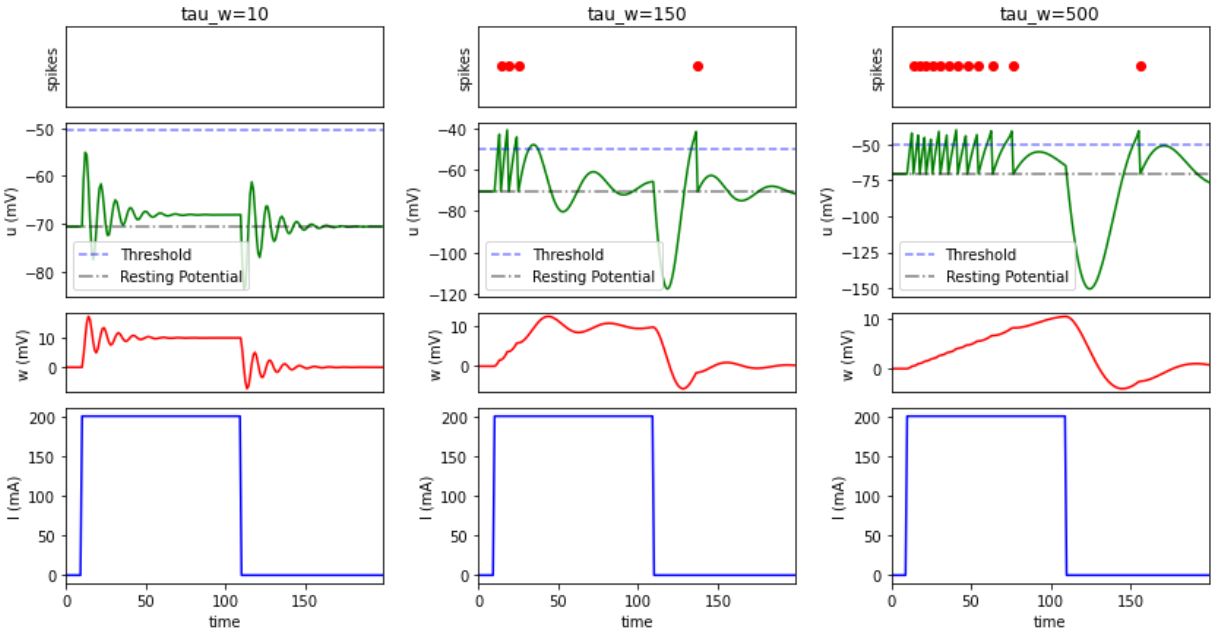
g.A.2. بررسی رفتار مدل با τ_wهای متفاوت

In [83]:

```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
Ion = 200
I = step_function(time, 10, val1=Ion) - step_function(time, time-90, val1=Ior

neuron_behaviour(AELIFPopulation, I, p, tau_w=10, postfix='1', name="tau_w=10
neuron_behaviour(AELIFPopulation, I, p, tau_w=150, postfix='2', name="tau_w=1
neuron_behaviour(AELIFPopulation, I, p, tau_w=500, postfix='3', name="tau_w=5

p.show()
```



نتیجه کاملا منطبق بر انتظار ۱۱ام ما از مدل است. در نمودار سمت چپ می‌بینیم زمانی که مقدار τ_w کم است، مقدار w به راحتی با تغییر پتانسیل نورون تغییر می‌کند و تقریبا با آن هماهنگ است. این یعنی adaptivity سریع و ضعیف! درحالی که از چپ به راست، بین نمودارها، مشاهده می‌کنیم با افزایش τ_w، لختی w بیشتر می‌شود؛ دیرتر تغییر پیدا کرده و دیرتر هم بازمی‌گردد.

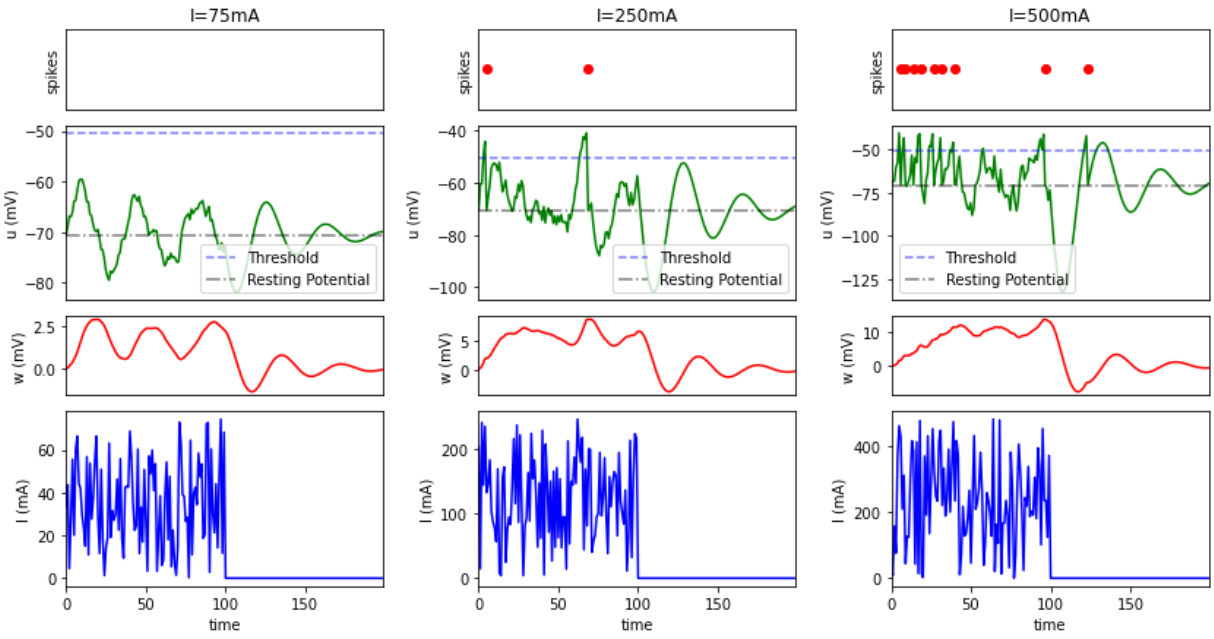
B.2. جریان ورودی تصادفی

a.B.2. بررسی رفتار مدل با دامنه‌های متفاوت جریان ورودی

In [96]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)

I = torch.rand(time,1)*75
I[int(time/2):] = 0
neuron_behaviour(AELIFPopulation, I, p, postfix='1', name="I=75mA", w=True)
I = torch.rand(time,1)*250
I[int(time/2):] = 0
neuron_behaviour(AELIFPopulation, I, p, postfix='2', name="I=250mA", w=True)
I = torch.rand(time,1)*500
I[int(time/2):] = 0
neuron_behaviour(AELIFPopulation, I, p, postfix='3', name="I=500mA", w=True)

p.show()
```



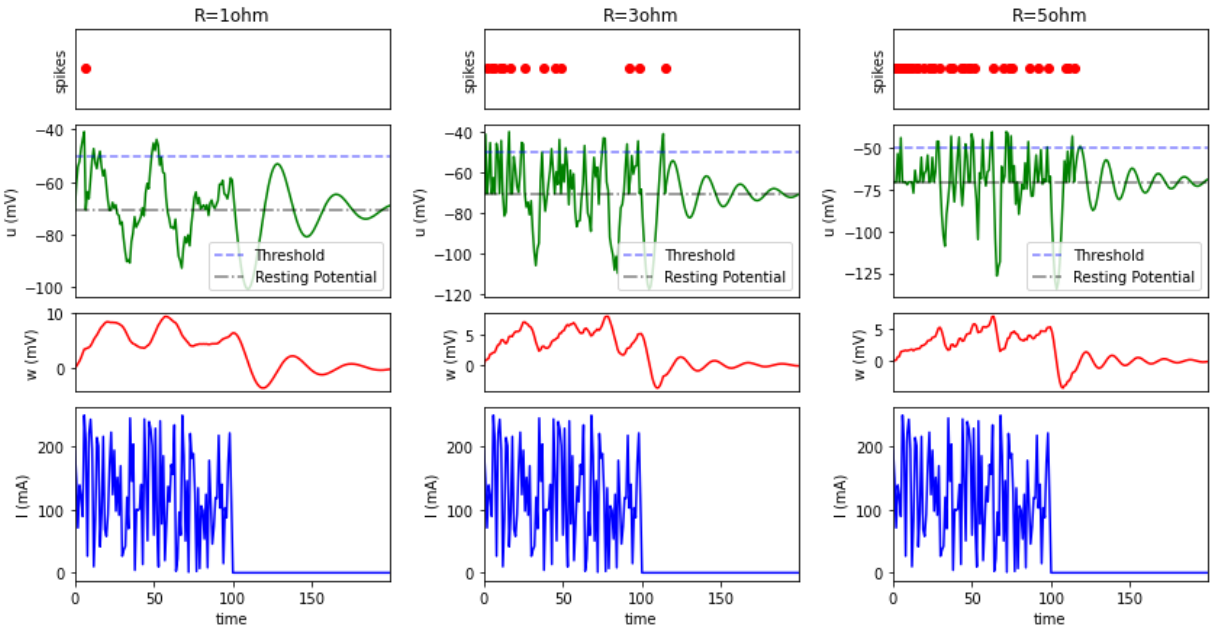
.مشابه آنچه در بخش متناظر برای ورودی ثابت گفته شد، نمودارهای بالا کاملا گویای انتظارات ۱،۷،۸،۹ و ۱۰ از مدل می‌باشند

b.B.2. بررسی رفتار مدل با مقاومت‌های متفاوت

In [97]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
I = torch.rand(time,1)*250
I[int(time/2):] = 0

neuron_behaviour(AELIFPopulation, I, p, R=1, postfix='1', name="R=1ohm", w=Tr
neuron_behaviour(AELIFPopulation, I, p, R=3, postfix='2', name="R=3ohm", w=Tr
neuron_behaviour(AELIFPopulation, I, p, R=5, postfix='3', name="R=5ohm", w=Tr

p.show()
```



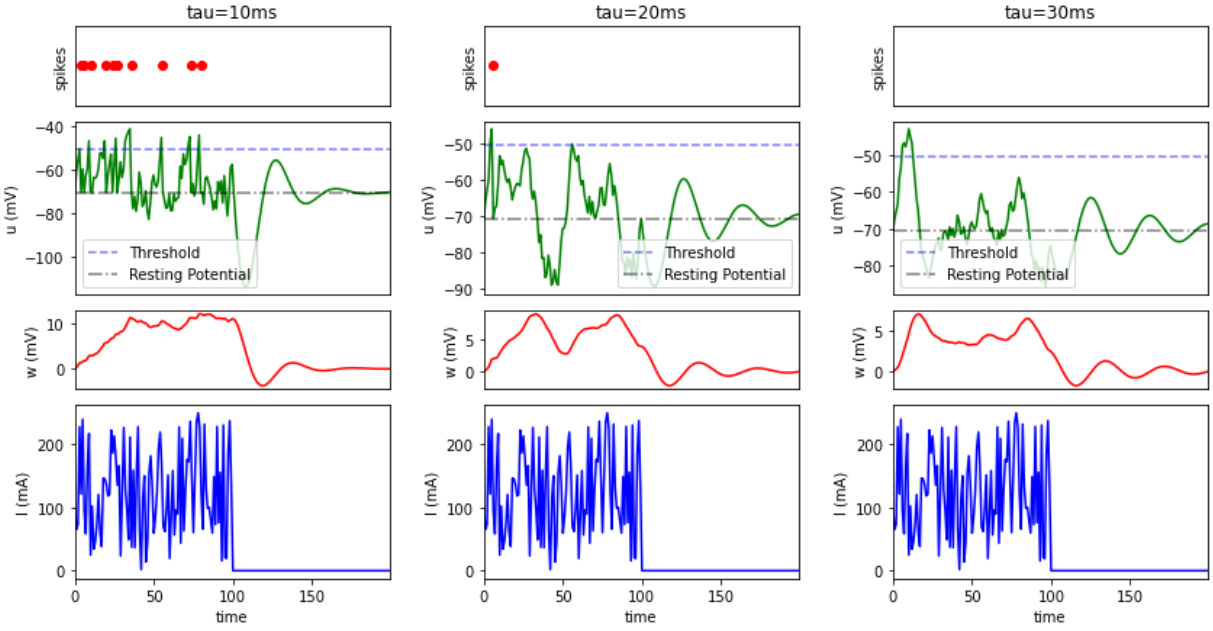
نتیجه مطابق انتظار دوم ما از مدل است.

c.B.2. بررسی رفتار مدل با ثابت زمانی(τ)های متفاوت

In [98]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
I = torch.rand(time,1)*250
I[int(time/2):] = 0

neuron_behaviour(AELIFPopulation, I, p, tau=10, postfix='1', name="tau=10ms",
neuron_behaviour(AELIFPopulation, I, p, tau=20, postfix='2', name="tau=20ms",
neuron_behaviour(AELIFPopulation, I, p, tau=30, postfix='3', name="tau=30ms",

p.show()
```



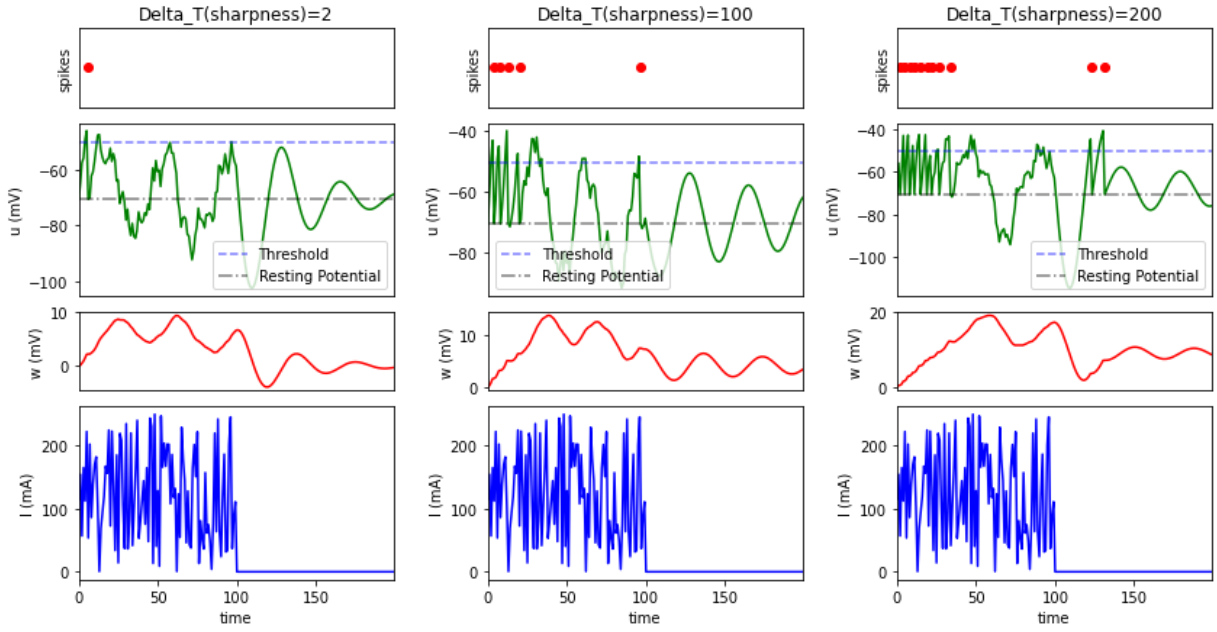
نتیجه مطابق انتظار سوم ما از مدل است.

d.B.2. بررسی رفتار مدل با Δ_Tهای متفاوت

In [99]:
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
I = torch.rand(time,1)*250
I[int(time/2):] = 0

neuron_behaviour(AELIFPopulation, I, p, sharpness=2, postfix='1', name="Delta
neuron_behaviour(AELIFPopulation, I, p, sharpness=100, postfix='2', name="Del
neuron_behaviour(AELIFPopulation, I, p, sharpness=200, postfix='3', name="Del

p.show()
```



نتیجه مطابق انتظار پنجم ما از مدل است.

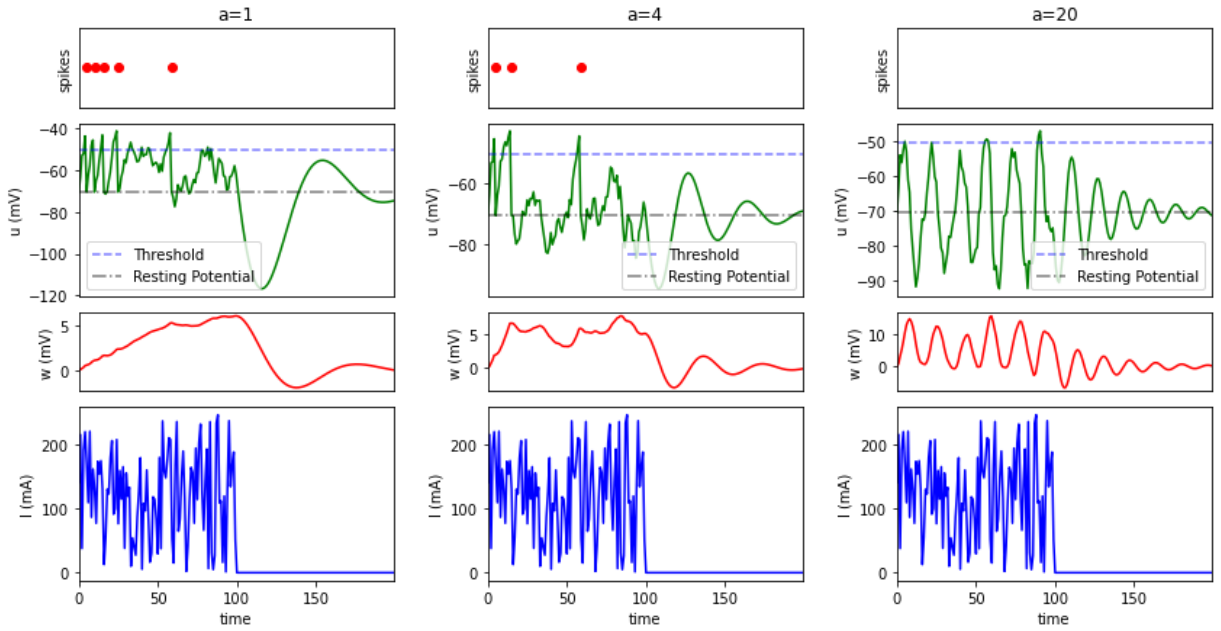
e.B.2. بررسی رفتار مدل با مقادیر متفاوت پارامتر a

In [100…
```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
I = torch.rand(time,1)*250
I[int(time/2):] = 0

neuron_behaviour(AELIFPopulation, I, p, a_w=1, postfix='1', name="a=1", w=Tru
```

```
neuron_behaviour(AELIFPopulation, I, p, a_w=4, postfix='2', name="a=4", w=Tru
neuron_behaviour(AELIFPopulation, I, p, a_w=20, postfix='3', name="a=20", w=T

p.show()
```



مشابه آنچه در بخش متناظر برای ورودی ثابت گفته شد، نتیجه مطابق انتظار ۱۳ام ما از مدل است.

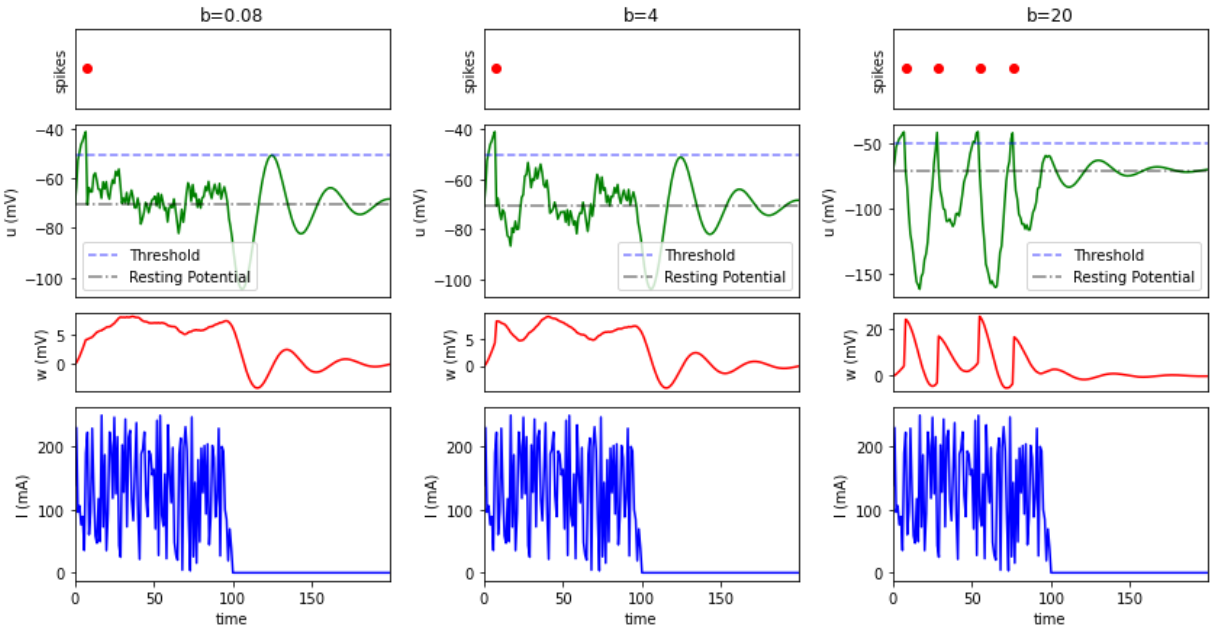
f.B.2. بررسی رفتار مدل با مقادیر متفاوت پارامتر b

In [101…

```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
I = torch.rand(time,1)*250
I[int(time/2):] = 0

neuron_behaviour(AELIFPopulation, I, p, b_w=.08, postfix='1', name="b=0.08",
neuron_behaviour(AELIFPopulation, I, p, b_w=4, postfix='2', name="b=4", w=Tru
neuron_behaviour(AELIFPopulation, I, p, b_w=20, postfix='3', name="b=20", w=T

p.show()
```



مشابه آنچه در بخش متناظر برای ورودی ثابت گفته شد، نتیجه مطابق انتظار ۱۲ام ما از مدل است.

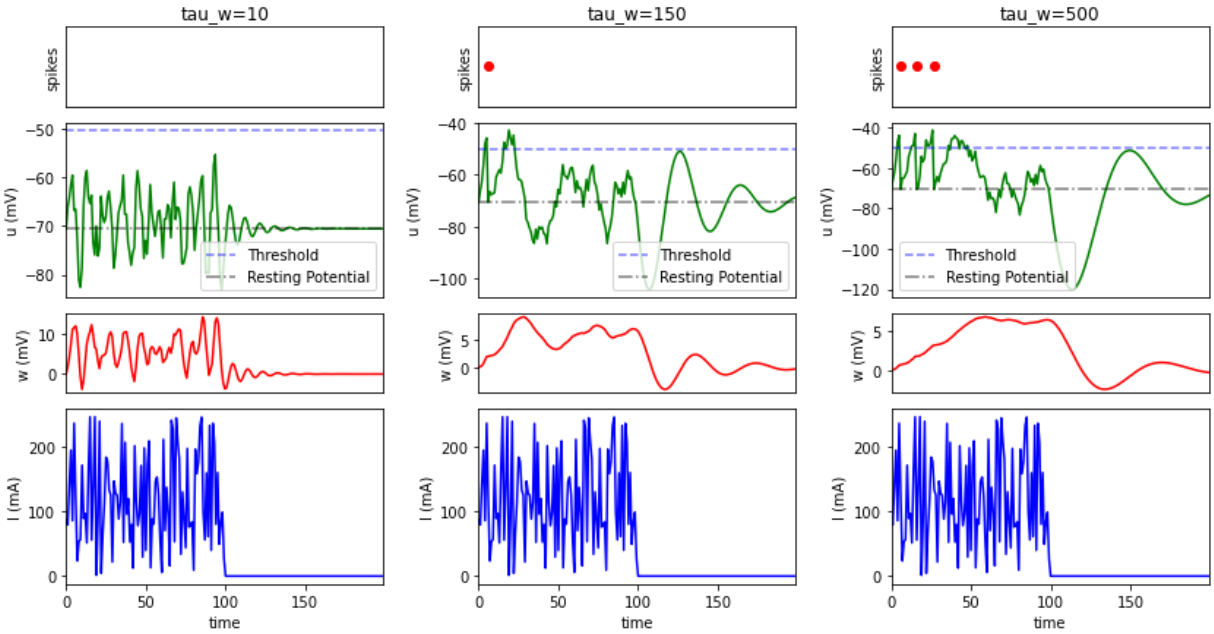
g.B.2. بررسی رفتار مدل با τ_wهای متفاوت

In [102…

```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
I = torch.rand(time,1)*250
I[int(time/2):] = 0

neuron_behaviour(AELIFPopulation, I, p, tau_w=10, postfix='1', name="tau_w=10
neuron_behaviour(AELIFPopulation, I, p, tau_w=150, postfix='2', name="tau_w=1
neuron_behaviour(AELIFPopulation, I, p, tau_w=500, postfix='3', name="tau_w=5

p.show()
```



مشابه آنچه در بخش متناظر برای ورودی ثابت گفته شد، نتیجه مطابق انتظار ۱۱ام ما از مدل است.

C.2. ورودی جریان پالس مربعی متوازن

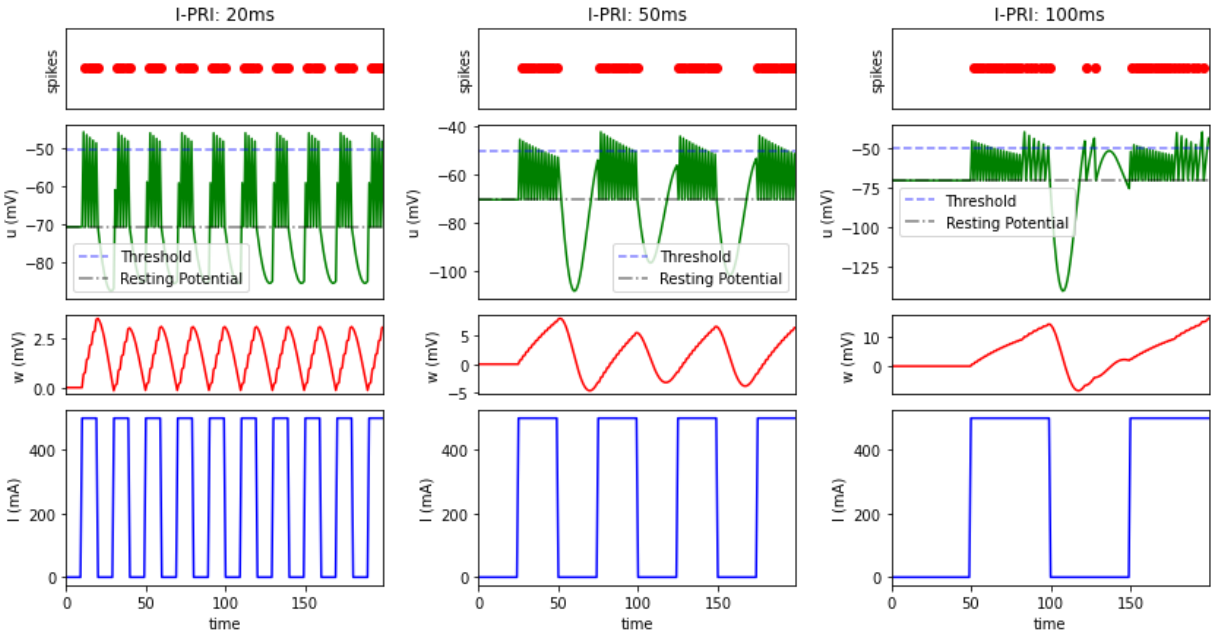
a.C.2. بررسی رفتار مدل با I-PRIهای متفاوت جریان ورودی

In [106…

```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
Ion = 500

I = [0]*10+[Ion]*10
neuron_behaviour(AELIFPopulation, I, p, postfix='1', name="I-PRI: 20ms", w=Tr
I = [0]*25+[Ion]*25
neuron_behaviour(AELIFPopulation, I, p, postfix='2', name="I-PRI: 50ms", w=Tr
I = [0]*50+[Ion]*50
neuron_behaviour(AELIFPopulation, I, p, postfix='3', name="I-PRI: 100ms", w=T

p.show()
```



می‌بینیم که ورودی غیر پیوسته باعث فعالیت بیشتر نورون می‌شود چرا که adaptivity نورون از کار می‌افتد (دلیل استفاده از صدای آژیر برای مواقع خطر)

b.C.2. بررسی رفتار مدل با τ_wهای متفاوت

پیشتر دیدیم که با کاهش τ_w، لختی adaptivity کاهش پیدا می‌کند. انتظار می‌رود در برابر ورودی غیر پیوسته (پالسی)، adaptivity با لختی کمتر تأثیر بیشتری داشته باشد و فرکانس spike خروجی را کاهش دهد. این اثر را بررسی می‌کنیم.

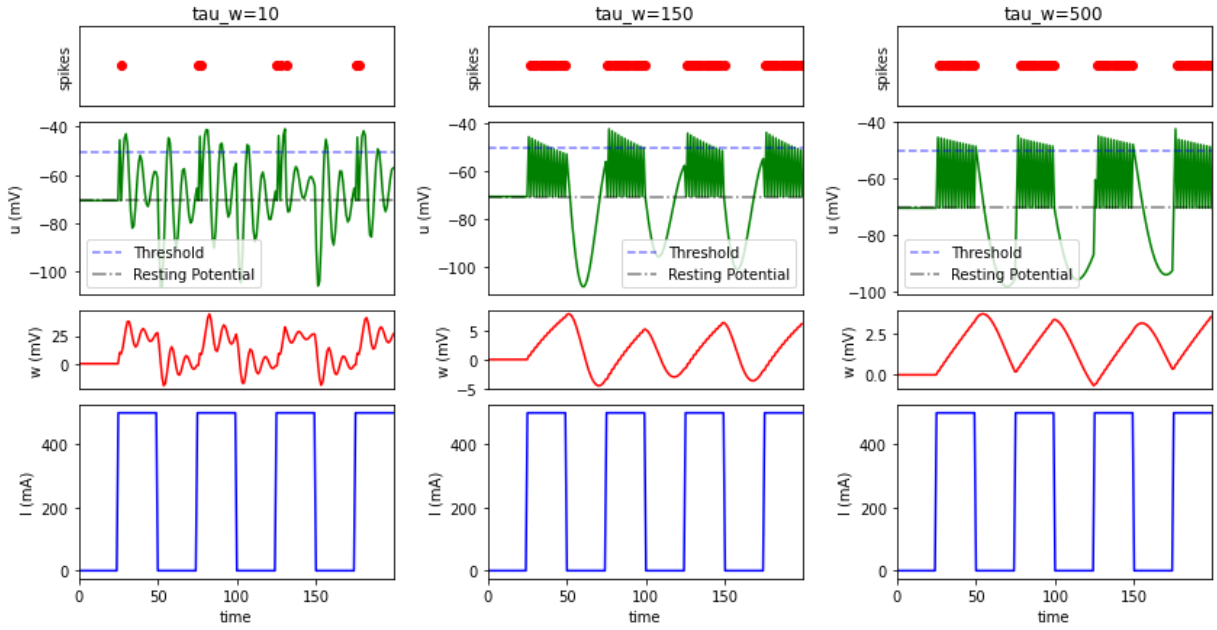
In [107…

```
plt.figure(figsize=(14,7))
```

```
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    ['w1','w2','w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
Ion = 500
I = [0]*25+[Ion]*25

neuron_behaviour(AELIFPopulation, I, p, tau_w=10, postfix='1', name="tau_w=10
neuron_behaviour(AELIFPopulation, I, p, tau_w=150, postfix='2', name="tau_w=1
neuron_behaviour(AELIFPopulation, I, p, tau_w=500, postfix='3', name="tau_w=5

p.show()
```



نتیجه مطابق انتظاری است که بالاتر بیان شد.

3. مقایسه مدل‌ها در کنار یکدیگر

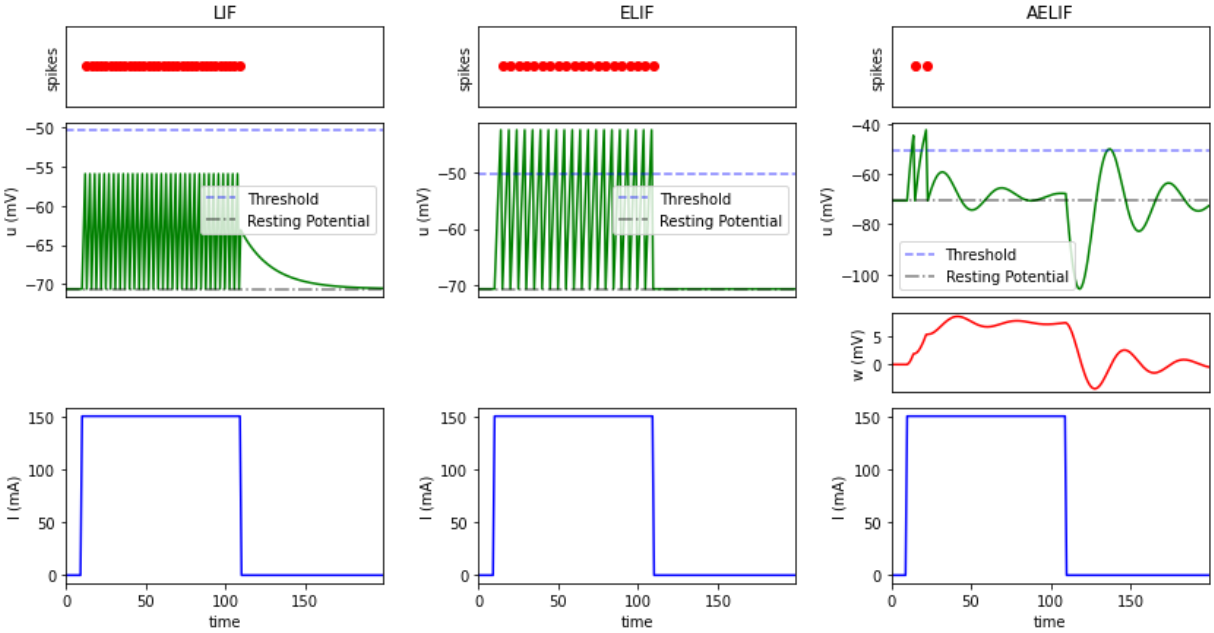
A.3. رفتار نورونی

In [115…

```
plt.figure(figsize=(14,7))
p = Plotter([
    ['s1','s2','s3'],
    ['u1','u2','u3'],
    ['u1','u2','u3'],
    [None,None,'w3'],
    ['i1','i2','i3'],
    ['i1','i2','i3'],
], wspace=0.3)
Ion = 150
I = step_function(time, 10, val1=Ion) - step_function(time, time-90, val1=Ior

neuron_behaviour(LIFPopulation, I, p, postfix='1', name="LIF", spike_threshol
neuron_behaviour(ELIFPopulation, I, p, postfix='2', name="ELIF")
neuron_behaviour(AELIFPopulation, I, p, postfix='3', name="AELIF", w=True)

p.show()
```



B.3. نمودارهای F-I

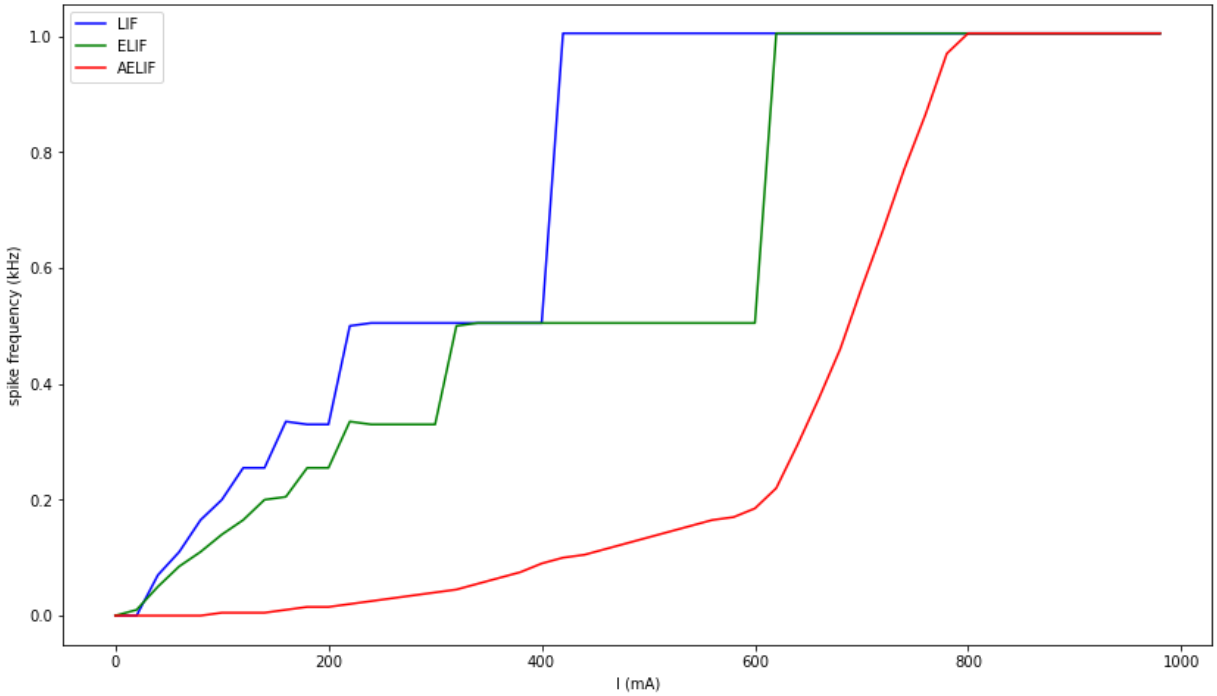
در این بخش، رفتار نورون‌ها را در برابر میزان شدت جریان ورودی (ثابت)، با استفاده از نمودار F-I مشاهده می‌کنیم.

In [118…

```
I_range = range(0,1000,20)
def cal_FI(neuron, I_range=I_range):
    monitor = Monitor(neuron, state_variables=["s", "u"], time=time)
    f = []
    for i in I_range:
        neuron.refractory_and_reset()
        I = torch.ones(time,1)*i
        monitor.simulate(neuron.forward, {'I': I})
        f.append(sum(monitor['s'])/time)
    return f

plt.figure(figsize=(14,8))
p = Plotter([
    ['F'],
], hspace=0.3)

neuron = LIFPopulation((1,), dt=1., spike_threshold=-50.4)
p.plot('F', y='F', x='I', data={'F':cal_FI(neuron,I_range), 'I':list(I_range)
       color='blue', y_label="spike frequency (kHz)", x_label="I (mA)", title
neuron = ELIFPopulation((1,), dt=1.)
p.plot('F', y='F', x='I', data={'F':cal_FI(neuron,I_range), 'I':list(I_range)
       color='green', y_label="spike frequency (kHz)", x_label="I (mA)")
neuron = AELIFPopulation((1,), dt=1.)
p.plot('F', y='F', x='I', data={'F':cal_FI(neuron,I_range), 'I':list(I_range)
       color='red', y_label="spike frequency (kHz)", x_label="I (mA)")
p['F'].legend()
p.show()
```



دلیل کمتر بودن spike frequency مدل AELIF نسبت به دو مدل دیگر واضح است. این کمتر بودن به دلیل کارکرد adaptivity است. اما کمتر بودن این معیار برای مدل ELIF نسبت به مدل LIF قابل توجه است. دلیل این کمتر بودن آن است که مدل ELIF مدت زمانی رو در دوره‌ی firing می‌ماند و پس از آن reset می‌شود که باعث می‌شود زمان کمتری برای spike زدن نسبت به مدل LIF داشته باشد.